Processing an Instrumental Gesture in the Reactable as a Method of Computer Improvisation

William Marley

MASTER THESIS UPF 2013

Master in Sound and Music Computing

Master thesis supervisor:

Sergi Jordà

Department of Information and Communication Technologies

Universitat Pompeu Fabra, Barcelona

Acknowledgements

I would like to express my deep gratitude to my family for their patience and support. This experience would not have happened without them.

I would like to thank my supervisor Sergi Jordà for his advice and guidance throughout the Master. Thank you to Xavier Serra for the opportunity to participate as an SMC student.

Finally, thanks to all of my classmates for being great classmates.

Abstract

Musical improvisation is one of the most widely practised forms of composition and performance. It is evident in most musical cultures and can be traced back to some of the earliest artistic and religious musical practices. It is a style that incorporates such concepts as change, adjustment, experimentation and ornamentation. Musical improvisation can demand cohesion and participation, a communicative and cooperative relationship between performers. We propose a method of capturing and reshaping an instrumental gesture on the Reactable, with the aim of establishing a computer improvisation system that will respond to the performer with variations on their contributed, physical gesture. A method of sampling this gesture, specifically the physical rotation of a tangible object, is detailed. We look to techniques used in Computer Aided Algorithmic Composition (CAAC) for a practical approach to data generation from a given sampled gesture in real-time. A stochastic procedure is designated as this algorithmic approach, one that generates new gesture data sequences from the performers original instrumental gesture. Bi-directional communication limitations within the Reactable are encountered and overcome in the construction of this system. A method of continuous play-mode activation based on a probabilistic routine is put forward. The testing of this system is described, with user responses and experiences discussed. Finally, plans for future development are outlined.

List of Figures

Contents

Chapter 1

1. Introduction

This thesis establishes a link between multiple disciplines of computer-music performance and composition in an effort to provide a methodology for the design of a computer 'co-player' or improviser. The objective is to create a bi-directional, 'call-and-response' relationship between the human performer and the computer-music system, one that upholds the users input 'style' while creating variations upon it. There is no single methodology that encompasses this sector of research, with many if not all other endeavours discovered throughout this thesis using separate, blended techniques. In addition to methods, there is a disparity in the tools used to accomplish the task, much in the same vein as a musician's preference for a particular instrument or system to accomplish a musical goal.

Relevant methodologies used in these separate cases will be described later in this thesis. In this chapter, it is necessary to highlight some relevant background of the previously referred to disciplines. The fundamental intention of design in this case is the generation of unique musical content given certain initial conditions and input. The purpose is to formalize the structure of the content producer, and as a result produce formalized musical content. The technique of applying formulaic rules to the creation of music is not a new one, with musical examples dating back centuries. In fact, the algorithmic approach to musical content generation has been practiced for thousands of years [Ed11].

However, only in the 20th century has it been given the name 'Algorithmic Composition'. It is nevertheless important to provide a brief historical background of the field before venturing into the technical aspects of these methods.

1.1 Algorithmic thought in Composition

It is, of course, impossible to pinpoint the earliest piece of algorithmically composed music. It is, also, not important to do so. Algorithmic thought in composition can be traced back to Pythagoras who believed in an inextricable link between mathematics and music, that music and numbers could not be separated. He theorized on the concept of 'Musica Universalis', and how the movement of celestial bodies are mathematically harmonious. He believed the positional relationship between the planets and the stars hold identical mathematical information to musical structures, intervals and even notes. Even though the music of these early Greeks was often improvised, it is an indication of the early musings on the formalization of the musical process.

In or around 1026, the Benedictine monk and inventor of modern musical notation Guido d'Arezzo set about formalizing a musical structure by assigning pitches to vowels in a given text. Compositions constructed from number ratios, for example Fibonacci numbers leading to the Golden Ratio, became a popular method around the 14th century. An example of this method is the motet *Nuper Rosarum Flores* by Guillame Dufay, composed for the consecration of Florence Cathedral in 1436. The timing structure of the motet is based on the ratios 6:4:2:3 – these ratios representing the proportions of the nave, the crossing, the apse and the height of the arch of the cathedral (some believe these ratios represent the proportions of the Temple of Solomon)[Ed11].

Introduced in the 18th century were musical dice games invented by composers like Johann Philipp Kirnberger, Maximilian Stadler and Joseph Hadyn. These dice games, or Musikalisches Würfespiel, were an early example of aleatoric methods of algorithmic composition. The most famous and cited of these dice games was composed by Mozart,

where a Minuet was created by choosing one of 8 pre-composed parts of a waltz depending on the result of a roll of the dice.

In more recent years, pioneers in the field of algorithmic composition began to flourish, the earliest being Joseph Schillinger (1895 – 1943). With his book the *Schillinger System of Musical Composition* (1946) he developed a mathematical and algorithm based method of musical composition. Schillinger invented algorithms for generating or transforming melodies or rhythmical structures, in addition to musical forms themselves. Such inventions and concepts have impacted more recent compositional practices, including the work of Karlheinz Stockhausen and Gottfried Michael Koenig [ECE07].

With the growing popularity and presence of computers during Schillingers period came a parallel growth and interest in algorithmic methods of composition, naturally due to the powerful number-crunching abilities of the developing computer processors.

1.1.1 Computer Aided Algorithmic Composition

The first experiments with computer-generated music took place in 1955 at the hands of Lejaren A. Hiller (1924-94) and Leonard M. Isaacson at the University of Illinois. The piece of music that resulted was entitled *The Illiac Suite for String Quartet (1956).* This piece was not performed by a computer; instead the musical material was created and calculated by the University of Illinois' ILLIACI (Illinois Automatic Computer) and the output transformed manually to musical notation. The piece consisted of 4 movements, which they called 'experiments'. They applied probabilistic algorithms to a three-step compositional procedure: a 'generator' which supplied initial material, a 'modifier' which applied transformations to the initial material and a 'selector' which selected material for output. Hiller went on the to develop MUSICOMP, one of the first computer languages developed for computer aided algorithmic composition (CAAC), and the multimedia piece *HPSCHD* with John Cage [ECE07].

The advent of these new computer-generated experiments and languages brought an influx of further experiments and projects in the field of CAAC. The extension of the serialist movement, invented by Arnold Schönberg (1874 – 1951) and sustained by his pupil Anton Webern (1883 – 1945), was bolstered by these new capabilities. Gottfried Michael Koenig's *Projekt 1* (1963) is an example of the first computer programs that generated whole musical structures in the serialist style.

The subject cannot be thoroughly discussed without addressing the work of Iannis Xenakis (1922 – 2001). Known primarily for his instrumental compositions and his work in architecture and engineering, Xenakis was also known for his innovations in algorithmic composition and computer music [Ed11]. He was highly critical of the serialist movement and it's musical output –

> *"Linear polyphony destroys itself by its very complexity; what one hears is in reality nothing but a mass of notes in various registers. The enormous complexity prevents the audience from following the intertwining of the lines and has as its macroscopic effect an irrational and fortuitous dispersion of sounds over the whole extent of the sonic spectrum."* [Xen55]

Xenakis coined the term 'Stochastic Music', a technique based upon random operations within time-variable constraints. This will be discussed in more detail further in this text. Xenakis went on to propose this principle of indeterminacy as a solution to the "crisis" of serialism –

> *"For if, thanks to complexity, the strict, deterministic causality which the neo-serialists postulated was lost, then it was necessary to replace it by a more general causality, by a probabilistic logic that would contain strict serial causality as a particular case. This is the function of stochastic science."* [Xen92]

Xenakis utilized the power of the computer to its utmost, developing the UPIC system (a computer based machine dedicated to interactive composition), implementing probabilistic works using the FORTRAN language running on an IBM-7090 in Paris (1965) and developing GENDY (GEN for generation, DY for dynamic), a program written in Basic that generated both the structure of the piece and the actual sound material used [Ser93].

1.1.2 Computer Aided Improvisation

It is beyond the scope of this thesis to undergo a complete study on the artistic form that is improvisation. It is relevant, however, to address the importance of improvisation as a musical style, and outline a number of developments in computer improvisation systems.

Improvisation is one of the most widely practised and least understood music styles in the world. It does not belong to any one culture or region, and is a technique found in music compositions and performances throughout history and amidst the multitudes of global cultures. It is a technique used both in religious veneration and artistic expression. It is involved in change, adjustment, development and experimentation. It requires participation and cohesion [Bai93].

George Lewis' *Voyager* system (1985) is an excellent example of a highly complex, deliberately personal and independent computer improvisation system. It is designed to perform at will with one or multiple performers in the style of free improvisation. This style of improvisation is in contrast with the system design proposed by this thesis, to the extent of their programmed independences and deliberately designed complexities. With *Voyager*, Lewis is addressing the concept of "multidominant elements" - which he terms "multidominance" – found in the musical and visual works of Africa and its diaspora [Lew00]. This notion is explored in *Voyager* with its propensity towards multi-rhythmic musical content played by several groups of ensembles - sometimes simultaneously, sometimes not – with its 64 asynchronously operating "players". Such complexity and

independence is designed with a highly intentional reference towards this concept of "multidominance" and free improvisation.

Voyager functions as "an extreme example of a "player" program, where the computer system does not function as an instrument to be controlled by a performer" [Lew00]. This follows Robert Rowe's taxonomy of the "player" and "instrument" paradigms [Row92]. Again, this functionality is in contrast with the system being proposed in this thesis project, as a level of control is desired in the 'call-and-response' designed relationship. The specific goals of this thesis will be clearly stated at the end of this chapter.

While Lewis conceived a performance with *Voyager* as "a non-hierarchical, improvisational, subject-subject model of discourse, rather than a stimulus/response setup" [Lew00], a system in the form of François Pachet's The *Continuator* [Pac03] is an example of a far more user-dependant, stylistically observant computer co-player. This system learns the users style from the input material presented to it. It employs the 'call-and-response' behaviour previously stated and applied in this thesis project. The *Continuator* responds with variations on the user input data by reading back portions of the data that may have suitable continuations given any starting pitch or subsequence [CE07].

These example systems are presented to highlight the diversity of approaches taken in the design of a computer improvisation system. The following chapter will contain more specific approaches taken and techniques used in this field of research, those of which are more closely relevant to this thesis.

1.2 The Reactable

The Reactable is an electronic musical instrument with a table-based tangible user interface (TUI) used for control and visual-feedback. The instrument is heavily inspired by the analogue modular synthesizers of the 60's [Jor10], and does in fact employ the same modular synthesis format by creating signal chains of multiple generator, effect and control modules. The Reactable functions with the use of acrylic pucks, representing

these modules, and their rotation, table-position and proximity to other connected modules. These pucks have been described as “building blocks of electronic music, each one having a different functionality in sound generation or in effect processing” [Jor10]. The idea of simultaneously playing and constructing the instrument is conceptually derivative of the visual programming environment that runs the synthesis ‘engine’ of the Reactable. The environment in question is Pure Data (Puckette 1997).

Whereas the analogue modular synthesizer used physical patching to create links between modules, the Reactable simply depends upon the presence (or not) of a module in an appropriate proximity to another to automatically produce these links. As more modules are linked in this chain, the potentiality of greater complexity is available to the performer. This is so due to the presence of multiple controllable parameters in each module, affording to the module the ability to substantially transform and manipulate the incoming signal.

The table’s surface is made from a translucent circular sheet of perspex. Each puck has a unique identifier in the form of a fiducial symbol located on the tableside face of the puck. The puck is then tracked by a camera that is situated underneath the table surface (inside the physical instrument), with its identifier and gestural behaviour data then acquired in the open-source computer-vision framework reacTIVision [KB07]. This software detects the position and rotation of the fiducial, in turn sending this data by means of the TUIO Protocol to the sound and graphical processors. Visual feedback is projected back onto the surface from underneath, with representations of the actual sound waves being output in real-time presented to the user. In addition to this, the table has a multi-touch feature, which couples with the rotation of the puck in controlling parameters.

The Reactable was conceived as, among other notions, an instrument for collaboration [Jor05]. One need only consider its circular shape to understand the potential, if not demand, for simultaneous, collaborative play. The Reactable in concert has been seen to function solely through improvisation, while also maintaining the capability of providing pre-determined, instrumental backing in large-scale group performances [Jor10].

1.3 Motivations

It is beneficial to the reader to highlight the main motivation for addressing this subject matter as being derived from personal experience, and a natural inclination towards improvised performance methods. As a self-taught musician, of various instruments and at varying levels of proficiency, the experimental approach to music creation was born not of choice but of necessity. The standard theoretical rules of composition and performance were not at hand during collaborative sessions or when attempting to compose material. Instead, my musical education came out of frequent and lengthy communal experimentations with musicians of the same ilk. During these sessions, it would not be uncommon to play non-stop for long periods, developing a theme out of initial incoherence and evolving this theme through embellishment and ornamentation. At times the result could be seen as nonsense, but other moments would conjure an unmistakeable understanding and musical bond between players that would be very difficult to recreate in a different situation. In fact, these improvisations would not only be very difficult to recreate, but most would be irretrievable with no possibility of resurgence. This seemingly somber facet was in fact a fundamental source of satisfaction; a unique event had taken place, with the elements that created that unique event becoming the focus, not just the event itself. Serendipity, a positive chance occurrence, was the goal here.

Improvisation is a style of music that, due to its unpredictable nature, can lead to new and fruitful ideas in the development of a piece. With the computers' gargantuan processing capabilities, it should be a foregone conclusion that improvisation systems be the focus of research. It will be seen further in the report that others have strived to create interactive computer-music systems specifically with these goals in mind.

In a very practical sense, one must take into account the availability of collaborators (or, in a lot of cases, a lack there of). Having personal experience with this situation, the idea of generating an artificial partner to perform with is striking as a very desirable ambition. This concept has an obvious link to current research in AI and HCI, both of which explore the nature and potential of the interactive and potentially creative relationship

between computers and humans. The author is of the opinion that it is an important route to take, and considering the nature of this project I feel it can aid in the development of computer-musicianship, and as a result, our own musicianship.

1.4 Goals

The goal of this research is to discover a methodology for employing computer improvisation techniques in an interactive computer-music system, more specifically, the Reactable. On a more basic understanding, the goal is to integrate a system where the user guides the computer in a pseudo 'call-and-response' scheme. The user should provide the system with a performance style and, upon request, the system should respond with variations on that style. The specific performance style in question for this system is an instrumental control gesture. The Reactable utilizes physical human gestures significantly, and one could say that the physical gestures are the main tool used to navigate a performance. The Reactable is, after all, a tangible user interface, using the users touch and movement to control its various elements.

We must distinguish between a physical, instrumental gesture (the type that is being dealt with here) and a gesture in musical content. It will be seen that research in the field of improvising interactive music systems has been taking place for a number of years. It can also be seen that gesture recognition and analysis has been prominent in more recent times. However, the computer-improvisation of human gestures is much less the subject of research. Therefore, this goal to process a physical instrumental gesture in order to generate a sense of intelligent cooperation will hopefully be a contribution to current and future research.

This system will also focus on the manipulation of Reactable objects whose duty is to alter timbre i.e. filters and effects. The reasoning behind this is to concentrate on musical facets that are less in focus when working in the algorithmic composition domain. The author is of the understanding that the majority of work in this field is dealing with pitch and rhythm generation and manipulation. Therefore, another primary goal of this thesis

is to work in the domain of timbre control while retaining all pitch and rhythmic qualities assigned by the performer.

This focus takes part inspiration from the concept of *Klangfarbenmelodie,* or sound-colour-melody [Ano11]. The concept was originally devised by Arnold Schoenberg in 1911. It is the process of distributing melodies through the instruments of an orchestra, where different instruments may be assigned single notes of the melody line. This would create melodic textures and complex timbre distributions. We will not focus too sharply on this however, as this thesis cannot claim use specific techniques in the style of *Klangfarbenmelodie*. The only relevance is in its overall concept, and this systems attention being placed on timbre transforming Reactable processes.

This system is aimed towards experienced users of the Reactable mostly, but experienced users of interactive music systems should find some musical value in it if there is value to be found.

Chapter 2

2. State of the Art

The purpose of this chapter is to outline the techniques and approaches that are directly relevant to this thesis. It will cover the description of algorithmic methods of musical generation that have been found most suitable for this type of project, a description of machine listening with example systems and an outline of a proposed method of musical gesture improvisation.

2.1 Stochastic Methods of Musical Generation

The transference of the improviser as collaborator aesthetic into the computer domain leads one to address notions of bounded indeterminacy and chance. A stochastic process employs these notions, which is why they are so in use by algorithmic composers. Xenakis' idea of 'Stochastic Music' was that of random operations within time-variable constraints, relating the principles of indeterminacy and statistical organisation of mass structures to natural processes in our environment [ECE07]. Stochastic algorithms function under the mathematical principles of probabilistic distributions, where an outcome is defined by a given probability. The simplest and most widespread procedure in implementing these probabilistic distributions is the use of probability lookup tables [MMH99]. These tables store values that correlate with the probabilities of the

occurrence of musical events. There are various forms of probabilistic distributions, here are but a few, described in brief:

- Uniform Distribution (Probability Density Function) – each outcome has an equal probability of success
- Linear Distribution – a line of increasing or decreasing probability given two limits
- Exponential Distribution – an exponential growth curve given two limits
- Bell-shaped Distribution – used to describe normal distribution

An example of a uniform probability function would be the rolling of a dice or the drawing of a card from a pack. In a musical context, a non-uniform distribution can be used simply by assigning probabilities to the choosing of notes in a scale. Each note can be given a conflicting probability, the only restriction being that they must add up to 1 when summed.

Collins [Col10] maintains a useful function for continuous distribution in frequency continua or glissandi, or the controlling of timbral features. This echoes Xenakis' own theories on the functionality of continuous and discontinuous sounds - "a multitude of short glissandi on strings can give the impression of continuity, and so can a multitude of pizzicato" [Exa09]. McAlpine [MMH99] asserts that probability distributions can act as efficient decision-making procedures as follows:

"The composer specifies a number, n, of possible outcomes. With each of these outcomes is associated a procedure that will be executed if that particular outcome is selected, and a real number between 0 and 1 that gives the probability that the outcome will be selected. Upon normalization, this probability distribution corresponds to a partition of the real line segment [0, 1] into n distinct regions, whose interval widths are the probabilities of the respective outcomes. A random-number generator is used to generate a real number between 0 and 1. The decision can then be made by observing in which of the n segments the number lies"

2.1.1 Markov Chains

Allocating probabilities to given outcomes is too simplistic a technique for use in a system that requires an essence of collaboration, a system that displays some form of context knowledge. A Markov Chain of order *m* is a discrete probability system in which it retains memory about the past *m* events. This is useful in musical applications where a particular theme or 'flow' can be maintained. A Markov chain deals with conditional probabilities that unfold by considering previous states, the number of which are known as the *order* of the chain. A chain which only considers its previous state is a 1st order Markov chain; a chain which considers its 2 previous states is known as a 2nd order Markov chain, and so on. Generally, a state-transition matrix represents an Nth-order Markov chain, corresponding to an N+1 dimensional probability table [MMH99]. Figure 1 shows a first-order Markov chain transition table. Previous states are listed vertically while transition states are listed horizontally.

	A	B	C
A	0.25	0.5	0.25
B	0	1	0
C	1	0	0

Figure 1. First-order state transition matrix [MMH99]

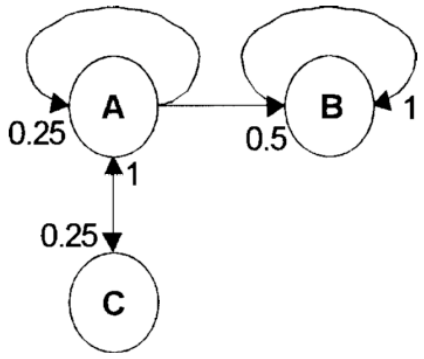

Figure 2. Labelled directed graph [MMH99]

Figure 2 shows McAlpines labelled directed graph for a first-order Markov chain. Focusing on this graph, state A and state C are said to be *reachable,* and therefore can *communicate.* States are shown to be *reachable* when it is possible to reach one state from the other in a finite number of steps. Again from this graph, neither A and B nor B and C *communicate*. This is so because neither state A nor state C is *reachable* from state B, which is always followed by itself [MMH99].

McAlpine claims that Markov chains are exceptionally well suited for rhythm selection, that rhythmic lines exhibit semi-cyclic behaviour in the form of repetitive short phrases. His CAMUS 3D generative system widely utilizes stochastic routines described above, "as they offer a quick and efficient method of specifying long-term structure while avoiding the often laborious task of specifying details at each step" [MMH99]. Eigenfeldt and Pasquier present a real-time chord sequence generation system that employs a unique user-influence over variable-order Markov transition tables [EP10]. Assayag et al engaged in experiments in *context modelling* with extensive use of Markov chains, based on the idea of complex sequences exhibiting a property called *short memory property* [ABC06][RST97].

Järveläinen [Jär00] can see problems relying on Markov chains as being the sole avenue of algorithmic composition for a particular project; "One of the problems with Markov chains is that unless the probabilities in the state diagram are generated randomly, they have to be derived from existing music. By such method the process generates an output of the same musical style as the input. That kind of music has hardly any compositional value." [Jär00]. However, it is because of this very fact that Markov chains are so useful, especially in improvisational systems. Many systems require an output that is similar in style, but not identical, to the input.

2.2 Machine Listening

The concept of the listening and reacting music machine edges us closer to the realm of artificial intelligence. There is a necessity now for electronic music, and by association interactive computer music systems, to be prepared to deal with improvised, spontaneous situations. We have yet to reach the limits of the processing and generational capabilities that machines can offer, with the onus placed on machines to be brought closer to our musical practices and behaviour [CE07].

What has been stated earlier is that there is no single method to create a computer music system that will improvise. Over recent years focus has been placed on the imitation of style: its storage, recall and alteration by a computer system [Pac03][ADD99]. Focus has also been aimed specifically towards human improvisers, creating computer improviser systems to produce embellished output to the input of the musician [Hsu05][Lew00].

Perhaps the best approach in dealing with these methods is a breakdown of the systems that relate to the project at hand.

2.2.1 William Hsu's Improviser System – William Hsu (2005, 2006)

Hsu's system focuses mainly on information gathered by timbral variations of live saxophone audio input. It was built for use in collaboration with British free-improvised saxophonist John Butcher. Specific goals were outlined from the start, with the principal goal being the systems ability to "analyse and respond to gestures with rich timbral information" [Hsu05]. This system is of particular interest in the context of this research due to its emphasis on "working with abstract sound, gesture, and texture, rather than more traditional parameters such as pitch and harmony." [Hsu05] The system records timbral features such as noisiness, inharmonicity of partials, sharpness of attack, presence of multiphonics, tremolo/flutter and audio roughness. Figure 3 shows a schema of the high level organization of the system. The audio input is fed to a set of audio analysis modules. The raw data is then processed to reveal rough descriptions of timbre

and other performance characteristics. These features and characteristics are monitored by an ensemble of improviser modules, with each module 'performing' based on a combination of user-commands, internal processes and extracted audio information.

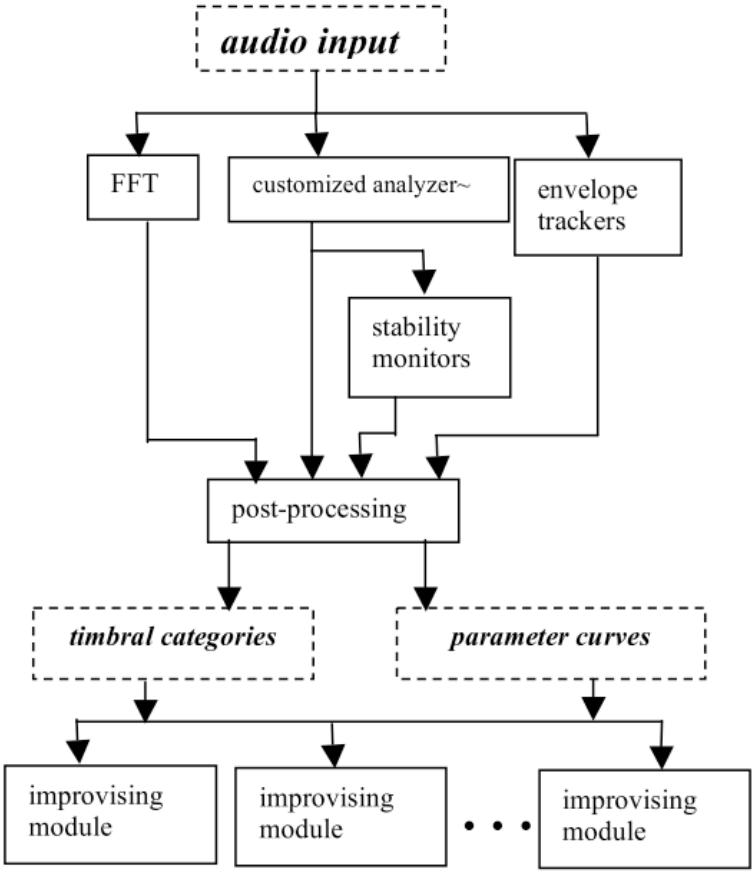

Figure 3. High-level organization of system [Hsu05]

The system operates with semi-autonomy, with the user retaining control of high-level operations if needed. However, the system can also function in 'auto-pilot', generating responses from a database of recorded gestural information. The system overall is highly customized to the specific style of Butcher, that being one of rich timbre variations.

2.2.2 Omax – Assayag et al (2006)

The Omax system is an improvisational musician-machine. It is an interactive system that learns in real-time from human performers. Omax's functional concept is based on sequence modelling and statistical learning, in a hybrid-architecture of OpenMusic and Max. Omax is easily one of the most versatile computer improvisation systems available (free), as it uses on-the-fly statistical learning for virtual improvisation generation and stylistic model archiving. The term 'Stylistic Reinjection' was coined in [ABC06] to describe the process of systematically re-sending mirror images of a performance back to a performer. Figure 4 shows a simple diagram of how this concept is visualized.

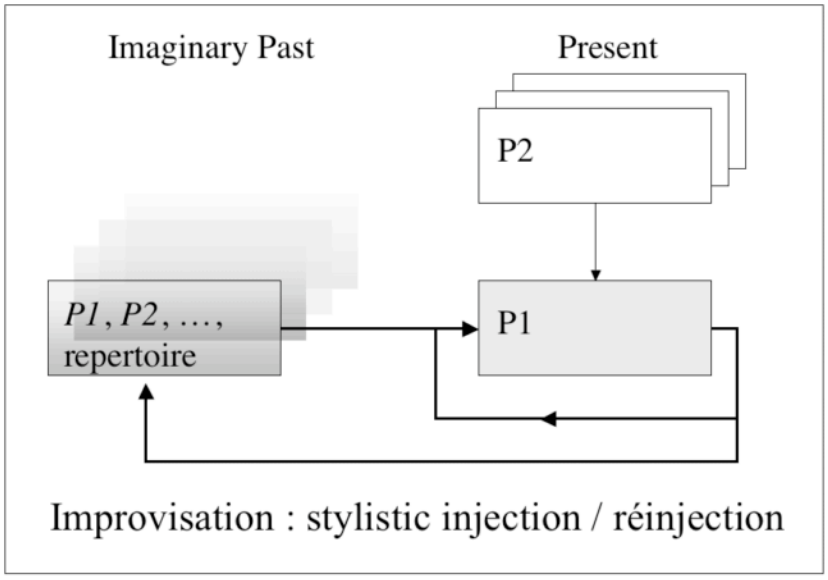

Figure 4. Stylistic Reinjection [ABC06]

With this feedback of musical data, the performer must react accordingly. Thus, in turn, the future of the performance is in constant flux. Within Omax, sound-images are memorized, stored as compressed models then reactivated as similar but not identical sequences. When building Omax, a particular guideline was set in regard to speed of learning. Learning must take place quickly and incrementally in order to come in-line with real-time interaction.

2.3 Instrumental Gesture Improvisation

In recent times there has been much research in the field of physical gesture capture, recognition and analysis. However, it is difficult to find research writings or developments in the area of instrumental gesture improvisation (post-capture). Christopher Dobrian of the University of California proposes a method of computer characterization of "gesture" in musical improvisation [Dob12], the definition of the gesture in question being that of "significant motion". This term does not bind the gesture in the physical domain, but implies that musical content can and does display significant motion. It satisfies the contention that musical gesture not only refers to the physical act but also to the musical content, that it implies motion that in turn characterizes meaning. Dobrian asserts that the "existing techniques used for tracking and analysing the physical gestures of the performer can therefore be applied similarly for tracking and analysing the gestural nature of the music itself" [Dob12].

It is the case that this thesis focuses on the physical, instrumental gesture of a musician performing on the Reactable. However, it was Dobrians insights into this topic of gesture characterization for improvisation that inspired the methodology for this thesis, in spite of the disparities between the two topics. The characterization of both gesture domains can be achieved with similar methods.

Dobrian mentions the mental techniques of "cross-domain mapping", the act of drawing correspondences between an incompletely understood domain and a useful target domain. In the case of gesture characterization, this mapping comes into effect when measuring and graphing any musical parameter over time. Essentially this can be used to visualize the morphology of the parameter over time in a two-dimensional space. In Dobrians case, this mapping is used to categorize and compare gesture 'shapes' in order to build a lexicon of gesture material for later recall.

In the case of this thesis, this mapping is useful for visualizing the instrumental gesture. The objective here is to visually analyse and compare the original gesture with the new, transformed one. It also supports the possibility of involving separate musical parameters for improvisation in future work.

Along with this mapping scheme, other insights proved worthy of note and relevant to this thesis. It must be stated again, however, that there are many differences in approaches to both topics. Dobrian's proposal is a more challenging one, with that of an automatic characterization of musical content and an intelligent response to it. This is reserved for future work in the case of this thesis.

Chapter 3

3. Methodology

The goal of this chapter is to describe in detail the methods and implementation of this new improvisation system. Some concepts previously discussed will be highlighted again, but only for context and in greater depth. There will first be a description of the Reactable software being used and its inner functionality. There will then be a discussion on the decisions made on the design of the new system, and how it could be integrated given the communication scheme and its limitations. The author will then discuss the construction of the new system, including the resolution of certain issues arising from the initial system limitations and the algorithmic processing involved in the construction.

3.1 Fitting in with Reactable

The main elements of the Reactable hardware have been discussed in chapter 1. What will be discussed here are the specific elements of the system that need to be considered for the design and construction of the improviser system.

As previously stated, the Reactable consists of a set of tangible objects that are placed on the table and controlled using instrumental gestures. The primary objects consist of sound generators, control generators and effects. These particular objects are defined as being ‘local’ objects, meaning they are linked to other objects in close proximity.

Secondary objects also exist, these being defined as ‘global’ objects. Their functionality is global in that they influence every process or object that is on the table at that particular time. They do not graphically link to any one object, instead they are invisibly linked to every object that is instantiated.

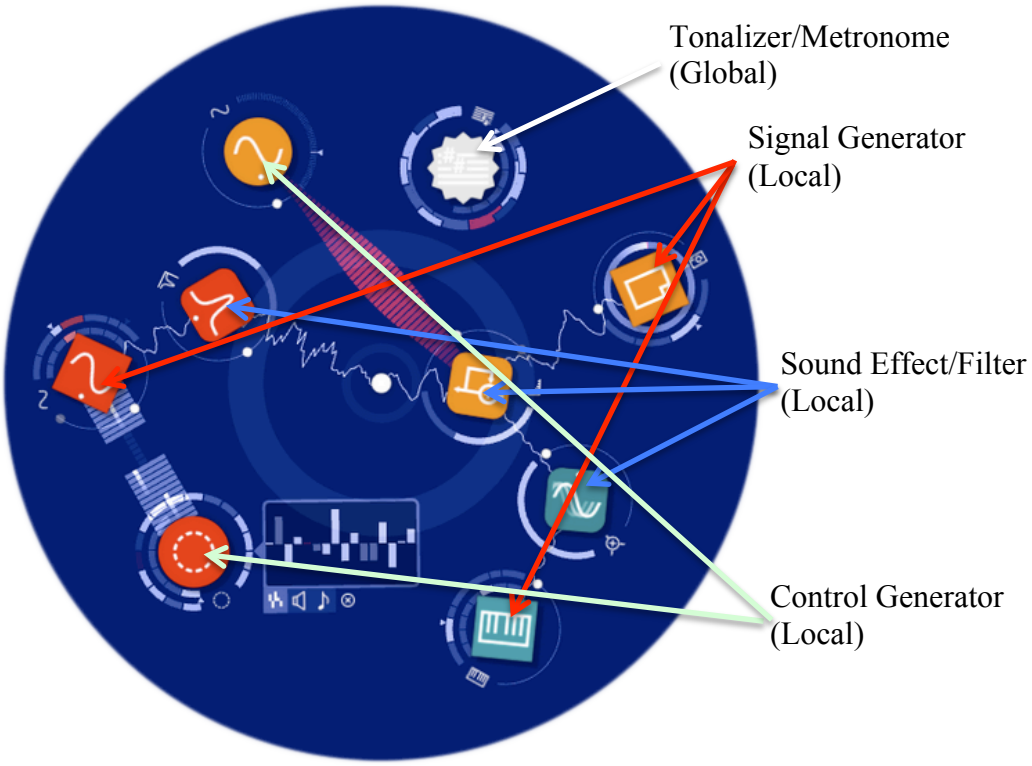

Figure 5. The Reactable with ‘local’ and ‘global’ objects

All sound functionality is coded in the Pd synthesis engine patch. It is here that communication is received from the graphical engine through the Reactable controller regarding object presence, movement and proximity to other objects. The synth engine will process these commands and act accordingly, telling each object what action needs to next be taken. Each object in the main synth patch is a dynamically loadable, independent patch. It relinquishes this dependence once an action command is given.

3.1.1 Design Considerations

Knowing this, certain decisions had to be made on how to integrate a computer improviser system into an already fully functioning interactive music system such as this. The first decision to make was an important one: introduce a new object or alter existing one? When addressing this issue it became apparent that the latter would be more problematic. It would be necessary to implement control over the rate of improvisation in the new system, so integrating this into already existing objects would alter their fundamental design. New control mechanisms would have to be put in place, more parameter options would need to be available, and an overcrowding of processes in a pre-designed object would hinder its common usage. It was also necessary to store gestures, with each gesture being easily accessible to the user. This would not be possible with a filter or effect, considering their simplistic parameter controllability. Therefore, the decision was an obvious one: a new object would have to be implemented.

After choosing to implement a new object, what kind of object should it be? Should it be global (control over multiple objects simultaneously) or local (portable, serves only single objects it is connected to). The problem of object differentiation arose when considering a global object; how does it decide which object to control? It would not be desirable to read gestures from one or more objects and reshape and replay them to all objects on the table. The local object opened a greater wealth of control opportunities. Its convenient and efficient table operations would be suitable for performance, with the ability to easily define which object to control and how.

A local control object would be created, one that would link to a filter or effect, record the gesture performed on this object, store that gesture, process for reshaping and playback to the same or another filter or effect.

Figure 6. ‘Gestroviser’

3.1.2 Communication Chain

It is useful here to highlight the signal and control chains, as they relate to the construction of the new system and identify some limitations in the Reactable as a whole. In figure 7. below, one can see the communication relationships between the primary objects.

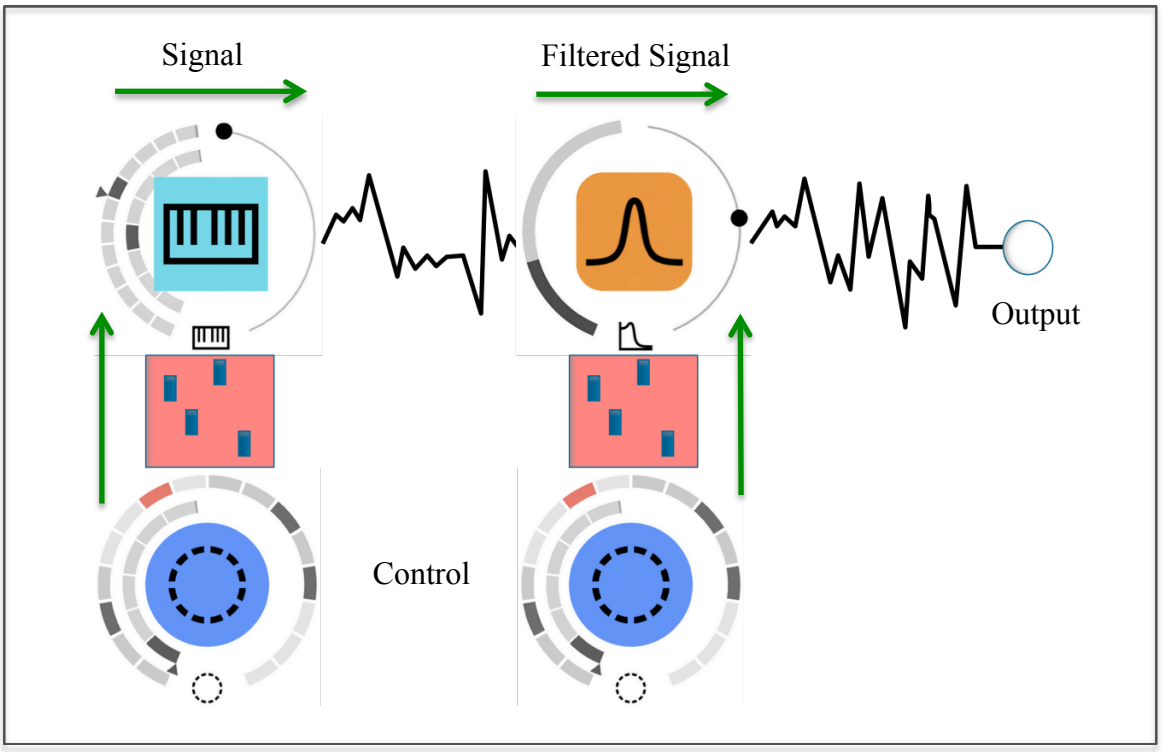

Figure 7. Communication relationship between primary objects

These relationships are of primary consideration when designing a system in order to augment the current processes. In figure 7. we can see the signal generator feeding the filter/effect module, which transforms the signal and sends it forward. This could be sent forward to another filter/effect in the chain, or simply sent to the output. One can also notice the direction of communication between the control object and both the signal generator and filter/effect objects. In all cases the communication is uni-directional. This brings us on to the limitations of the Reactable system.

3.1.3 System Limitations

The Reactable processes are very much similar to that of Pure Data itself in that it functions by linking modules together. They are both very much indicative of the way an analogue modular synth functions also. One feature of this modular functionality is the dependency on a uni-directional data flow. The communication scheme of the Reactable is designed so as to link each object in a forward communication path, but not backward. The main reason for this, and in other modular schemes, is the danger of the feedback loop. If a signal sends from one object in a forward link to another, and with the signal being sent back from the other to the origin, then a loop would occur and overload the system.

Figure 8. shows a diagram of the new improviser module, from here on named Gestroviser. This outlines the communication routine necessary for functionality.

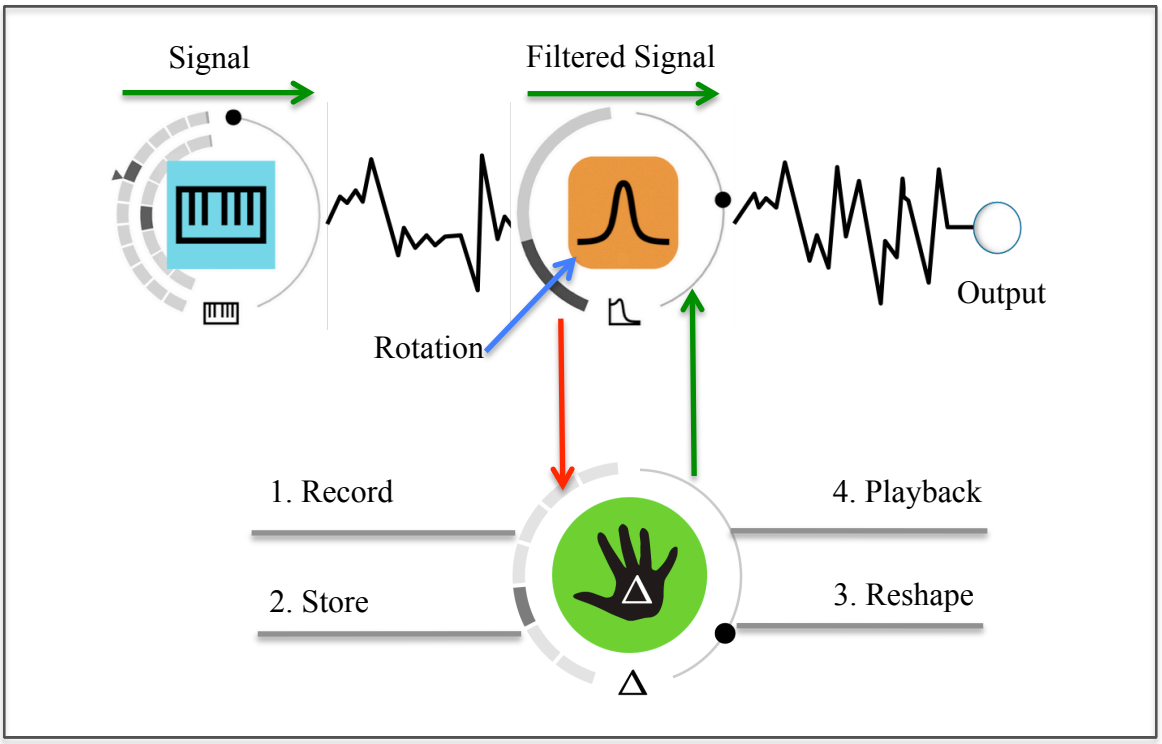

Figure 8. Gestroviser schema – bi-directional communication

Firstly, the Gestroviser needs to be able to connect automatically to the filter/effect modules. The Gestroviser then has to 'record' the physical gesture being applied to these objects. The gesture in question is the physical rotation of the object. It would then have to 'playback' this rotation gesture to the filter/effect. In order to achieve this bi-directional functionality, two 'channels' of communication have to be put in place (as can be seen from figure 8., red and green). This is a special case in modular systems due to the previously stated feedback issue. However, the Gestroviser could not function without this alteration. To state in more basic, conceptual terms, the computer must be able to 'listen' to the users actions. From there, the computer should process these actions, and playback these processed gestures under the supervision of the user.

3.2 Implementation

This section describes how these issues were solved, and how the system processing was achieved.

3.2.1 Building the Gestroviser

The Gestroviser framework is based on the original 'Melody' module in the Reactable Experience (Pd Reactable). The basic skeleton is exactly the same: 8 storage banks and a finger slider. Within this skeleton all of the processing would take place. The two objects are also similar in that they are both control modules, therefore certain conventions must be satisfied in both patches.

Being a control module, it must be first instantiated as text in files devoted solely to the objects and their initial settings. Once the new object and these settings are configured, a new graphical object exists which is necessary to bridge the communication path between graphics and synth engine.

3.2.1.1 Bi-directional Communication

Within the Reactable environment various conventions are in place. In order to link the physical, and in turn graphical movements of the objects to the synth engine there are certain abstractions one needs to be aware of. The Reactable patch has a parent patch called [main] where all other patches and abstractions are housed. In order to solve the bi-directionality problem a series of changes had to be made within the x_cconns abstraction. This is a helper abstraction that sends OSC commands to the Pd environment from the graphical engine. This tells the synth engine which object should attach to another, sending object IDs and their behaviour. This is where the commands needed to be intercepted and altered to suit our needs.

As can be seen from figure 9., the [x_cconns] abstraction informs object with ID 1 to attach to ID 2 automatically. In order to return communication back to ID 1, the process was doubled then reversed, essentially saying, "when ID 1 attaches to ID 2 then ID 2 must attach to ID 1".

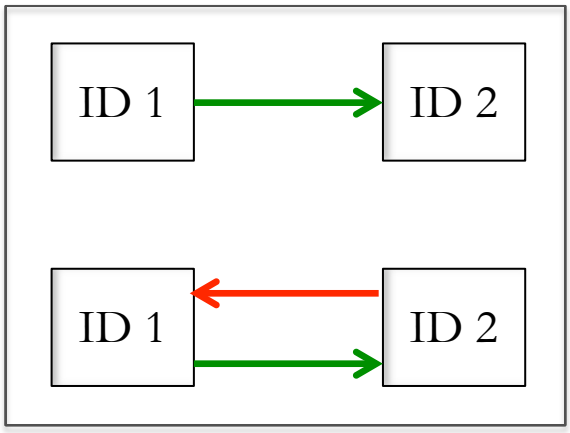

Figure 9. Creating the backward connection

The issue of data feedback was not a problem in this case. As will be discussed in the following section, the recorded data from the filter is sent on a separate stream as the playback. Playback is instantiated at a different time than record, therefore there is no overlap of data. Certain error messages were observed with 'confused' objects trying to link to other objects they shouldn't.

3.2.1.2 Data Routing

Of course, each object that required this backward connection needed to be altered so as to open the lines of communication. Not all objects in the Reactable are designed to do this. For example, when we focus on a filter, we can observe that only audio signals can pass through. It is not a control object, therefore it does not send out control messages i.e. messages that a control object can work with. The objective here is to instigate the passing of gesture data, in the form of control data, from the filter object while leaving the audio signal unaffected. Therefore certain flags needed to be installed in each object, specifically [$1 rot], so both objects would know precisely what to expect and where to deliver it.

3.2.1.3 Record and Store

Once the automatic backward linking problem was solved, it was necessary to setup the recording process. The manifestation of the gesture data is basically a stream of numbers. Therefore, a sample rate for the gesture needed to be considered. A rhythmic nature of each gesture was assumed. It was assumed that a gesture would usually follow a rhythm or beat, especially on an instrument like the Reactable which has generally been seen to be used for beat oriented electronic music. So the decision was made to activate recording on a beat, and last for 8 beats at a given tempo of 120bpm. Various sample rates were experimented with. For example, a rate of one gesture sample per beat was attempted, with a view to perform a type of interpolation between samples on playback. Another sample rate of 25 samples per beat was applied, with an overall range of samples at 200. This specific rate is currently preferred, but could be open to change in future work. The benefit of activating the recording procedure on a beat means that on playback (which is also activated on a beat) the gesture remains in time with the original intent of the user. Future possibilities of gesture quantization are currently being considered.

The gesture record object has 8 storage banks. As the object is rotated, a countdown of 8 beats begins and is visible on the table surface. This signifies that recording has begun. As the recording is activated, the 'path' for the data stream is open and unhindered by the recording process. After the countdown has finished, and thus the rotation gesture, playback automatically begins. As the playback function is activated, the communication path that facilitated the recording procedure is closed, un-hindering the playback. While playback continues, the user can switch between banks playing back the various gestures that have been recorded.

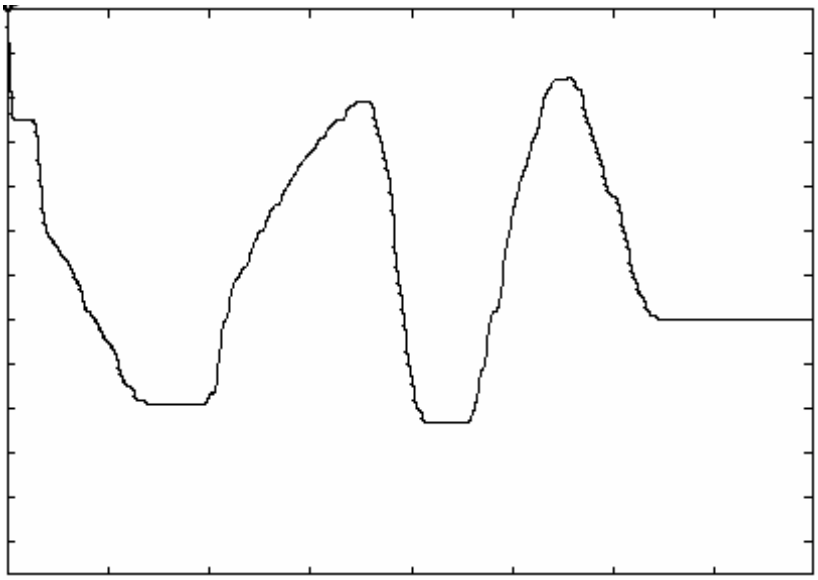

Figure 10. Recorded gesture – taken from Pd

3.2.1.4 Algorithmic Processing

In an attempt to have greater manipulation of the sampled gesture for processing, a routine was put in place to segment the gesture at sample time. With these segments came greater opportunity for transformation and manipulation before playback.

The sampled gesture would be split into 8 segments, coinciding with the 8 beats of the recording countdown. As previously stated, a strict adherence to a global syncopated beat was preferable. Each segment would then constitute 25 samples.

There was much deliberation previous to this stage regarding the choice of algorithmic technique to be applied to the sampled data. Thinking about the concept of improvisation, that being one of bounded indeterminacy, the decision was made to apply some stochastic procedures to the data. It was necessary to use an algorithmic process that could be achieved quickly, in real-time (or performance time).

For each gesture segment, a state-transition matrix (STM) would be created with a view to constructing a 2nd order Markov chain out of the sampled data. The idea behind this was to generate new data sets out of the gesture data, but sets that were based on the previously generated (user) data. Figure 11. shows the flow of data inside the Gestroviser patch. Starting with the rotation of a filter/effect, the data is split between the unaffected data and the segmented/processed data. The section following this will describe the play modes and how they are chosen for playback.

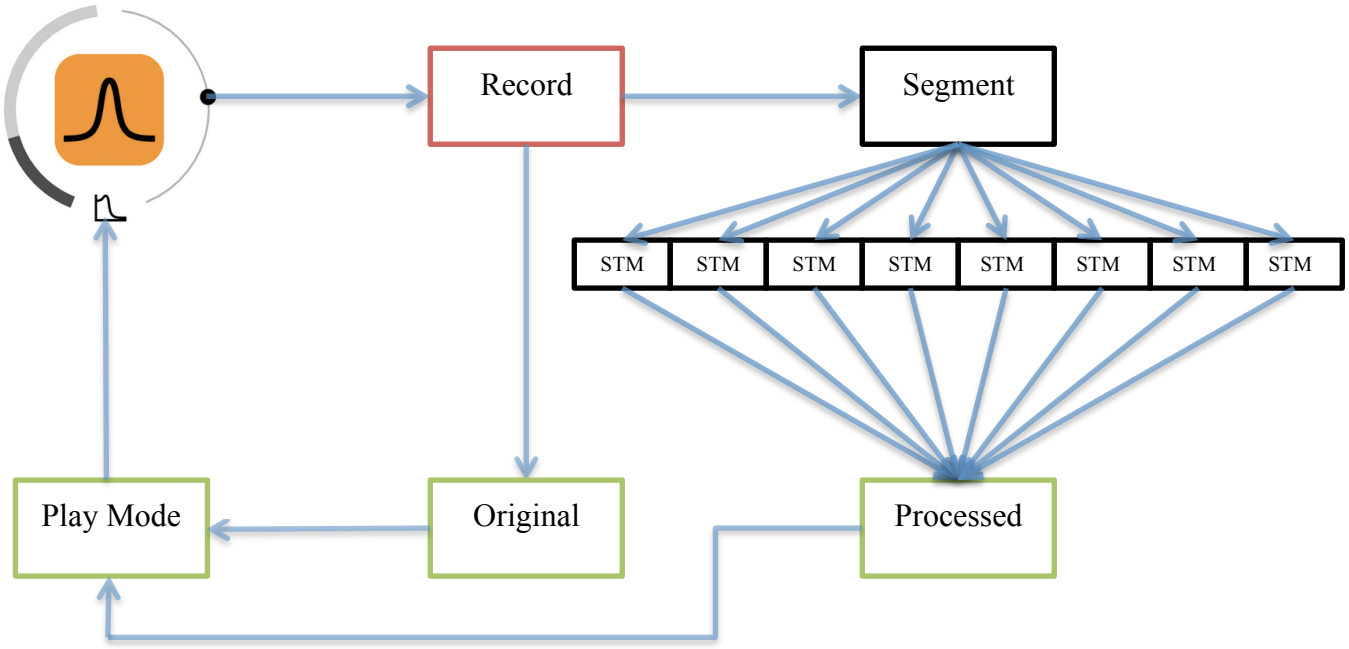

Figure 11. Data Flow inside the Gestroviser

A 2nd order STM is created by pairing couples of incoming gesture data for use as indices. These indices represent previous 'states', in this case the 2 previous states. The STM is then filled by registering all sampled points that would occur after these states. Therefore, one could see many sampled points occurring after a particular state, while also noticing only one sampled point occurring after a another event. The probabilistic process then occurs by choosing randomly from the list of possible points after each index event. This is not completely random, however (random processing is not the goal here). In a specific list there could be, for example, 10 points having occurred. Out of these 10 points, 5 of these could be the same point eg. 50. Therefore, the point (and therefore new event) 50 would have a .5 or 50% chance of being chosen from that list.

After each new STM is generated, they are then ready for immediate playback. On playback, each STM is read through with newly generated sampled points. Due to the use of probability in choosing output points, variations on each generated gesture is likely.

As can be seen from figure 11., the newly generated gesture shape holds an overall similarity with the original (in figure 10.). What can also be seen here are sharper peaks, changes from certain points to others. Initial thoughts on this were negative, with the

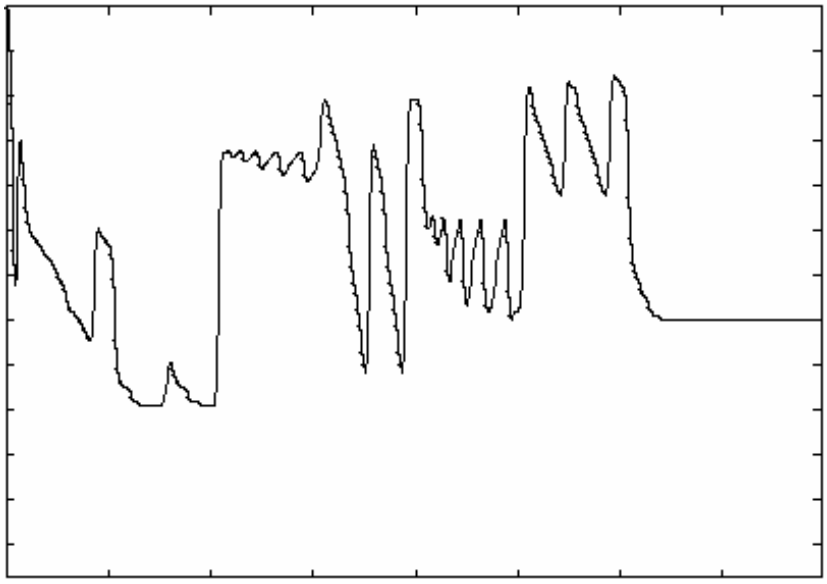

Figure 12. Generated gesture – taken from Pd

general feeling that this would produce an overly 'artificial' sense of computer response. However, if variation on a theme was a goal then this kind of behaviour could be harnessed. A smoothing procedure was experimented with, but in the end not implemented as a method of controlling the smoothing factor could not be decided on by test-time.

This technique is most commonly used for pitch generation [Ano12]. However, initial experiments using this procedure provided interesting results. The improving and augmenting of this method will be discussed in the next chapter.

3.2.1.5 Playback Modes

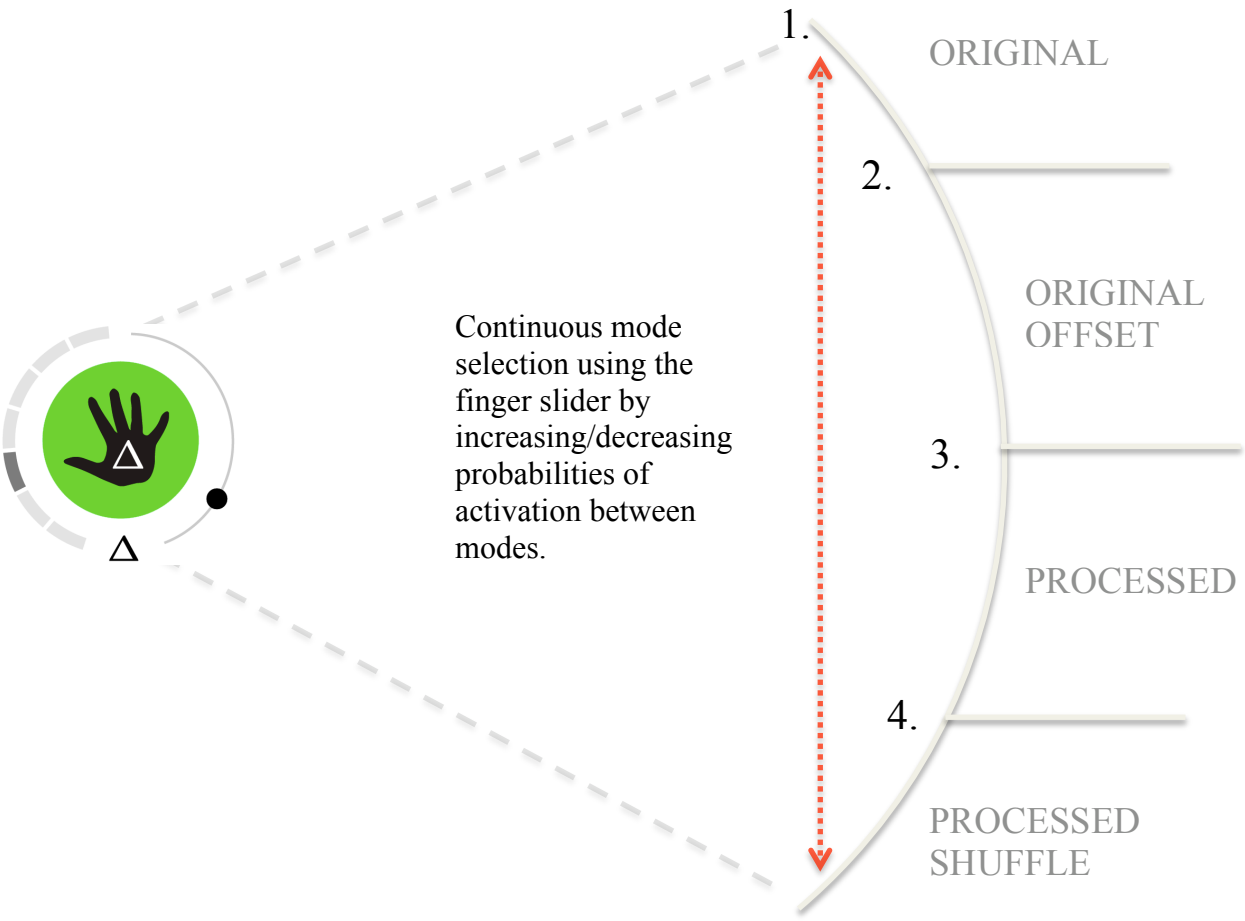

Figure 13. Play modes from finger slider

A method of choosing different playback modes was devised. Instead of using a discrete choosing operation, an alternative method was needed using the finger slider feature of the Gestroviser object. A procedure was put in place where the control of the slider would increase and decrease probabilities of the activation of each play mode. As before, the mode activation was syncopated with the global timing scheme. Therefore, a new mode (or current mode) is chosen on every beat.

To be more precise, it is useful to look to figure 13. for clarity. As can be seen, there are 4 different modes available. At the highest point, marked 1., the original gesture has 100% probability of being chosen for playback. As the slider moves downward, this probability decreases and the probability of the next mode becoming active increases. As the slider reaches the point marked 2 on the diagram, the probability of mode 2 (the original gesture but offset by 2 segments) is 100% with mode 1 at 0%. This procedure carries on through all modes and back again.

As previously stated, activation occurs on each beat. Therefore, if the slider is situated between 2 modes then there is a 50% chance either mode will be chosen. This procedure is applied to enhance the variability of output, while still resembling the original gesture. The table below gives a brief description of each mode.

Original	This is the original recorded gesture
Original Offset	This is the original, but with offset playback time (-2 beats)
Processed	This is the post-segmented, STM generated gesture
Processed Shuffle	Same as above, with random shuffling of segments

Table 1. Playback Modes

Chapter 4

4. Testing and Discussion

The following section describes the testing process. The Gestroviser system is a fully functional prototype, therefore test subjects were sourced to use it and answer pre and post test questions. This section will begin with an outline of the initial test criteria. The questions included in the questionnaire will be highlighted, along with the subjects' answers regarding their experiences. Following that will be a discussion about these experiences, and some musings on future work.

4.1 Testing Criteria

As outlined in the goals section of this report, this system is aimed towards experienced users of interactive computer music systems. Therefore, this had to be the main criteria when sourcing subjects to test the system. It would have been counter-productive to allow someone without any experience to test it, as most of the time spent during testing would be used by them to get there bearings on the Reactable itself before using the Gestroviser. This would be too time consuming and far from the point of the exercise. Therefore, it was an established fact before finding subjects that the amount of people available to test would be low.

Along with this, the decision was made to hold certain facts about the system from the test subjects before their testing of it. This was in order to judge the speed and extent of

their understanding of the system. Could they know what was happening straight away? If not, how long would it take them to understand? Is the system intuitively playable without prior knowledge?

It was important to realize that after working on this system for a long period of time, it would be impossible to disconnect oneself and make impartial judgements upon its usefulness. This realization is a main cause for the withdrawal of pre-test information from the subjects.

While the subjects would test the system, they would also be observed. This was simply to judge observed intuitiveness, enjoyment, frustration, confusion etc. The outcome of this observation was unknown, but was found to be quite useful at test time. In addition to written answers and comments regarding the system, some conversation post-testing took place. All significant observations and comments will be highlighted later in this chapter.

Before testing, the subjects were given basic instructions on how to use the Gestroviser. The instructions refrained from telling them what would happen when performed a certain function. Instead, they were told to apply the object to either of the 2 filter objects provided. They were told not to connect the object to the signal generator. They were also told that the rotation of the Gestroviser would achieve nothing (the gesture storage functionality was disabled for this test), so there would be no need to use that function. In addition, the subjects were told that using the finger slider of the filter would not be useful in this test. The reason for this was that the Gestroviser would only record the rotation data, no other gestural information. This fact was not supplied to the subjects, however. The instruction was simply not to use it. Again, this withdrawal of specific information was deliberate, the results of which will be seen in the next section.

4.2 Questionnaire: Questions and Answers

The following are the pre-test questions posed to the test subjects. The questionnaire was created in the Likert style, with subjects required to answer one of the following:

1. Fully disagree
2. Disagree
3. Indifferent
4. Agree
5. Fully agree

In the interest of efficiency, and due to the anonymity of the answers to each question, the percentages corresponding to each answer will be outlined after each question. The Gestroviser was tested using 5 test subjects.

1. I have experience making laptop and/or electronic music (20% = 3., 80% = 5.)
2. I am familiar with the Reactable and how it works (20% = 2., 20% = 3., 40% = 4., 20% = 5.)
3. I tend to improvise when making music (20% = 2., 40% = 3., 20% = 4., 20% = 5.)
4. I employ experimentation when making music (20% = 3., 60% = 4., 20% = 5.)

It can be noticed here that one test subject was not so experienced with the Reactable, interactive music instruments, improvising when making music or experimentation. Therefore the test would be particularly unfamiliar to them.

The following are the post-test questions posed to the test subjects, and again the percentage answer for each question is outlined.

1. I knew what was happening (60% = 2., 40% = 3.)
2. Give a very brief description of what was happening (n/a)
3. I was in control of what was happening (60% = 2., 40% = 3.)
4. I enjoyed using the test object (20% = 2., 20% = 3., 20% = 4., 40% = 5.)

5. I think this feature is useful for music making and/or performance (40% = 3., 60% = 4.)
6. I was aware of what the finger slider function was doing (40% = 1., 60% = 2.)
7. I found it useful to use the test object on both filters (20% = 2., 60% = 3., 20% = 4.)
8. Feel free to comment on your experience (supplied below)
9. Specific suggestions (if any)? (supplied below)

The following are comments made by some of the subjects when prompted to supply them. Firstly, when commenting on their experience:

Subject A – "*I think* [it] *is interesting to be able to use the same rotation recorded and apply it to different objects."*

Subject B – *"Really liked the idea of the computer being able to sensibly improvise Liked how some of the characteristics are retained in successive loops, but at the same time there are small changes, which give a sense of surprise This also makes the listener focus more on the musical nuances that keep changing throughout the performance The ring modulator wasn't as obvious as the filter Would be interesting to see the same idea with different control objects/tone generators."*

Subject C – *"I wasn't very sure about what was happening, and what would be happening when changing anything. I would probably have liked more clear/explicit visual feedback"*

Subject D – *"Throughout this experiment, I was mostly engaged in figuring out what the test object was doing rather than exploring my creativity with it."*

Secondly, when supplying specific suggestions:

Subject A – *"Maybe some labeling text appearing as you move the controllers (as it happens with the counter during the loop recording)."*

Subject B – *"more visual feedback (optionally) more "activity" (more & faster output density), when reusing the recorded gestures."*

Subject C – *"more instruction to be given prior to the experiment in terms of the functionality of the test object and its purpose."*

4.3 Discussion

After reviewing these results, some issues are clearly evident. The most evident of these is the fact that the subjects found it difficult to know what was happening with the Gestroviser. With all subjects answering "Indifferent" and "Disagree", it is clear that the functionality of the object is not so obvious. This was no surprise however, as this could be seen during the observation of their performance. A level of confusion was noticeable when observing certain subjects usage. On the other hand, enjoyment was observed in a different user, who appeared to understand the process and begin to utilize it in its intended fashion.

From the answers that were supplied, and the small amount of subjects present, it is quite possible to attribute these responses to particular subjects observed. For example, only one subject answered, "Fully agree" to the question regarding their penchant for improvisation. This, it could be argued, signifies that improvisation is their main form of musical creativity. This subject could be pinpointed during observation. At times this subject would apply chaotic and speedy movements to all objects, not in a fashion that implied a lack of knowledge of the instrument, but in an expert style that showed a familiarity with the instrument. Following on from this, the subject would supply the filter with small rotation gestures and wait for the Gestroviser to respond. After the recording of the gesture had taken place, the Gestroviser responded with a gesture of the same, low-density style of the original. This, to the observer, is a natural function of the Gestroviser; if it is given low-density material to process, it will respond with low-density material. This is not necessarily the ideal function of the Gestroviser however. It

was evident to the observer that this subject was an experienced improviser, and with the nature of the Gestroviser being an improvisation device then it should be expected to act in an improvisational style. This point is a digression towards a discussion on the different styles of improvisation i.e. free improvisation, variations on a theme etc. This, however, is quite beyond the scope of this thesis, and one on hold for future work.

It is quite clear that more graphical feedback is necessary in order to create more of an understanding of the Gestroviser. This was a mostly neglected aspect of the project, as more focus was placed on the inner capturing and processing of the Gestroviser. However, if a user cannot understand the process from the beginning then the processing side will be ignored.

The author gained more insight into some of the subjects' experiences during conversation after the test. One particular subject admitted to very little experience with electronic music, especially using interactive music systems. They experienced confusion during the test, then asking for an explanation of the system when the test was completed. When an explanation was supplied, the subject asked to use it for longer without being tested. With the knowledge of the process involved, they were visibly more confident, indicated a greater enjoyment when using it and generally made better use of the Gestroviser.

4.4 Future Work

Some technical points have been discussed in the previous section when outlining the responses of the test subjects. Here, the author will outline specific technical issues relating to this system that would need alteration and/or correction. Following that will be a broader description of future work on this topic.

Technical issues to be rectified:

- Stabilize connectivity scheme of the Gestroviser i.e. work with the graphics engine to disconnect Gestroviser from non-functioning objects
- Develop the graphical feedback process – the user needs to know precisely what is going on from the beginning, a more complete record/playback indication routine is necessary
- Enable multiple storage feature – this is desirable in order to record and store multiple gestures for future processing and playback
- Enable finger slider recording – project in its current state only records rotation data. A greater diversity of instrumental gesture, which could also include object movement, is desirable
- Adjust processing algorithm for potentially higher-density of output – this was a feature highlighted by a test subject, and observed by the author, a greater depth of system response is necessary

This thesis is a small step towards a greater goal that is the research and development of a more extensive and intelligent improvisational interactive computer music system. There has been much insight into the development of a basic system for capturing and processing a single instrumental gesture. What is necessary going forward is to think about more instrumental gestures using alternative computer instruments, most likely with the development of an instrument with this sole purpose in mind. With this, it is necessary to research methods to achieve a wider range of system responses. What has been seen throughout this research, specifically looking to the work of others, is that a more comprehensive learning scheme is an imperative. A system with a longer memory, an ability to develop a greater understanding of the user and their performance style and a ‘musical brain’ that can provide diverse and interesting musical material is an ideal blueprint for progressing in this topic.

References

[Ed11] Edwards, M. 2011. "Algorithmic Composition: Computational Thinking in Music." *Communications of the ACM*. http://dl.acm.org/citation.cfm?id=1965724.1965742.

[ECE07] Essl, K. 2007. "Live electronic music. *The Cambridge companion to electronic music"*, 107 - 125.

[Xen55] Xenakis, I. 1955. "La crise de la musique sérielle", *Gravesano review 1* (in: *Kéleutha. Ecrits,* L'Arche, Paris, 1994, pp. 40-2)

[Xen92] Xenakis, I. 1992. *Formalized music: thought and mathematics in composition*(No. 6). Pendragon Press.

[Ser93] Serra, M-H. 1993. "Stochastic composition and stochastic timbre: Gendy3 by Iannis Xenakis." *Perspectives of New Music*, 236-257.

[Bai93] Bailey, D. 1993. *Improvisation: Its nature and practice in music*. Da Capo Press.

[Lew00] Lewis, GE. 2000. "Too Many Notes: Computers, Complexity and Culture in Voyager." *Leonardo Music Journal* 10: 33–39. http://www.mitpressjournals.org/doi/abs/10.1162/096112100570585.

[Row92] Rowe, R. 1992. "*Interactive music systems: machine listening and composing*." MIT press.

[Pac03] Pachet, F. 2003. "The Continuator: Musical Interaction with Style." *Journal of New Music Research* (September). http://www.tandfonline.com/doi/full/10.1076/jnmr.32.3.333.16861.

[CE07] Collins, N. 2007. “Live electronic music. *The Cambridge companion to electronic music”*, 171-184. Cambridge University Press.

[Jor10] Jordà, S. 2010. “The Reactable: Tangible and Tabletop Music Performance.” *CHI 2010: 28th ACM Conference on Human Factors in Computing Systems,* http://dl.acm.org/citation.cfm?id=1753903.

[KB07] Kaltenbrunner, M, and Bencina, R. 2007. “reacTIVision: a Computer-vision Framework for Table-based Tangible Interaction.” *Proceedings of the 1st international conference on Tangible and embedded interaction. ACM, 2007*. http://dl.acm.org/citation.cfm?id=1226983.

[Jor05] Jorda, S. 2005. “Digital Lutherie: Crafting Musical Computers for New Musics’ Performance and Improvisation.” *PhD Diss Universitat Pompeu Fabra, Departament de Tecnologia (2005)*. http://mtg.upf.edu/node/449.

[MMH99] McAlpine, K, Miranda, E. and Hoggar, S. 1999. “Making Music with Algorithms: A Case-study System.” *Computer Music Journal*:19–30. http://www.mitpressjournals.org/doi/pdf/10.1162/014892699559733.

[Col10] Collins, N. 2010. *Introduction to computer music*. Wiley. com.

[Exa09] Exarchos, D. 2003. “The Sieves of Iannis Xenakis” *Mathematics and Computation in Music*. Springer Berlin Heidelberg, 2009. 419-429.

[EP10] Eigenfield, A, and Pasquier, P. 2010. “Realtime Generation of Harmonic Progressions Using Controlled Markov Selection.” *Proceedings of ICCC-X-Computational Creativity Conference*. 2010.http://www.metacreation.net/content/projects/01-Metacreation.dir/01-Kinetic_Engine/pdf/Controlled_Markov_Selection_(2010).pdf.

[ABC06] Assayag, G, Bloch, G. and Chemillier. M. 2006. “Omax Brothers: a Dynamic Yopology of Agents for Improvization Learning.” *Proceedings*

of the 1st ACM workshop on Audio and music computing multimedia. ACM, 2006. http://dl.acm.org/citation.cfm?id=1178742.

[RST97] Ron, D, Singer, Y., and Naftali T. 1997. "The Power of Amnesia: Learning Probabilistic Automata with Variable Memory Length." *Machine Learning* 25 (2-3): 117–149. doi:10.1007/BF00114008. http://link.springer.com/10.1007/BF00114008.

[Jär00] Järveläinen, H. 2000. "Algorithmic Musical Composition Algorithmic Musical Composition." Signal Processing (2000) Volume: 3, Issue: 1995, Publisher: Online article, Pages: 11

[ADD99] Assayag, G, Shlomo Dubnov, and O Delerue. 1999. "Guessing the Composer's Mind: Applying Universal Prediction to Musical Style." *Proceedings of the International Computer Music Conference*. 1999.http://articles.ircam.fr/textes/Assayag99a/index.pdf.

[Hsu05] Hsu, W. 2005. "Using Timbre in a Computer-based Improvisation System." *Proceedings of International Computer Music Conference* (5-9)

[Dob12] Dobrian, C. 2012. "A Method for Computer Characterization of 'Gesture' in Musical Improvisation" Christopher Dobrian Professor of Music." *Proceedings of the 2012 Computer Music Conference, Ljubljana, Slovenia.*

[Ano13] Anonymous, 2012. Second Order Markov Chains in Pure Data, [Web log post] Retrieved May 1st 2013